Programowanie w C# - podstawy

Programowanie w C# (**C-Sharp**) to zadanie, które można wykonywać przy użyciu różnych narzędzi. Oto kilka narzędzi często używanych w programowaniu w C#:

1. **Visual Studio:**
   - *Opis:* Visual Studio to jeden z najpopularniejszych zintegrowanych środowisk programistycznych (IDE) używanych do programowania w C#.
   - *Link:* Visual Studio
2. **Visual Studio Code:**
   - *Opis:* Lekki edytor kodu, który jest również bardzo popularny w społeczności programistycznej. Chociaż to nie jest pełnoprawne IDE, doskonale nadaje się do wielu zastosowań, zwłaszcza dla projektów na platformie .NET Core.
   - *Link:* Visual Studio Code
3. **.NET SDK (Software Development Kit):**
   - *Opis:* .NET SDK zawiera narzędzia, kompilatory i biblioteki potrzebne do rozwijania, kompilowania i uruchamiania aplikacji na platformie .NET.
   - *Link:* .NET SDK
4. **LINQPad:**
   - *Opis:* LINQPad to narzędzie, które pomaga w testowaniu i eksperymentowaniu z zapytaniami LINQ (Language Integrated Query) w C#. Jest to przydatne dla programistów pracujących z bazami danych.
   - *Link:* LINQPad
5. **Resharper:**
   - *Opis:* Resharper to narzędzie do analizy kodu, refaktoryzacji i dostarczania sugestii do poprawy jakości kodu w środowisku Visual Studio.
   - *Link:* Resharper
6. **Git:**
   - *Opis:* Git to system kontroli wersji, który jest szeroko stosowany w środowisku programistycznym do zarządzania kodem źródłowym.
   - *Link:* Git
7. **NuGet:**
   - *Opis:* NuGet to menedżer pakietów dla platformy .NET. Umożliwia łatwe dodawanie, usuwanie i aktualizowanie bibliotek w projekcie C#.
   - *Link:* NuGet
8. **Postman:**
   - *Opis:* Postman to narzędzie do testowania API, które umożliwia wysyłanie żądań HTTP, testowanie odpowiedzi i debugowanie interakcji z API.
   - *Link:* Postman

Te narzędzia są tylko kilkoma przykładami, a wybór zależy od Twoich preferencji i potrzeb projektu. Warto eksperymentować z różnymi narzędziami, aby znaleźć te, które najlepiej spełniają Twoje oczekiwania.

Kompilator do języka C# jest dostarczany jako część środowiska .NET od firmy Microsoft. Domyślny kompilator C# nosi nazwę **`csc.exe`** (C# Compiler).
W przypadku używania narzędzi takich jak Visual Studio, kompilacja jest zazwyczaj automatycznie zarządzana, ale można również skorzystać z wiersza poleceń.

Aby skompilować plik źródłowy C# za pomocą wiersza poleceń, możesz użyć następującej składni:

csc.exe

Na przykład, jeśli masz plik źródłowy o nazwie "MojProgram.cs", możesz skompilować go za pomocą polecenia:
csc.exe MojProgram.cs

Po wykonaniu tego polecenia zostanie utworzony plik wykonywalny o nazwie "**MojProgram.exe**", który można uruchomić.

Pamiętaj, że środowisko .NET oferuje wiele więcej narzędzi związanych z kompilacją, zarządzaniem zależnościami i budowaniem projektów. Jeśli pracujesz nad większym projektem, zalecam korzystanie z narzędzi takich jak MSBuild lub korzystanie z zintegrowanych środowisk programistycznych, takich jak Visual Studio.

Visual C# to część środowiska programistycznego **Visual Studio**, które jest rozwijane przez firmę Microsoft. C# (C-Sharp) jest językiem programowania stworzonym przez Microsoft, a Visual C# to narzędzie, które dostarcza zaawansowane funkcje edytora, debugowania, projektowania interfejsu użytkownika i zarządzania projektem do programowania w języku C#.

Oto kilka kluczowych cech Visual C#:

1. **Inteligentny edytor kodu:** Oferuje automatyczne uzupełnianie kodu, sugestie poprawek, nawigację po kodzie i wiele innych funkcji ułatwiających pisanie i zrozumienie kodu.
2. **Debugowanie:** Zapewnia zaawansowane narzędzia do debugowania, umożliwiające śledzenie kodu, ustawianie punktów przerwania i analizę zmiennych podczas wykonywania programu.
3. **Projektowanie interfejsu użytkownika (UI):** Umożliwia projektowanie interfejsu graficznego użytkownika za pomocą funkcji WPF (Windows Presentation Foundation) lub Windows Forms.

4. **Zarządzanie projektem:** Ułatwia tworzenie, zarządzanie i kompilację projektów w języku C#. Obsługuje różne typy projektów, takie jak aplikacje konsolowe, aplikacje okienkowe, usługi internetowe itp.
5. **Integracja z platformą .NET:** Visual C# ściśle współpracuje z platformą .NET, co umożliwia korzystanie z bibliotek klas, frameworków i narzędzi dostępnych w ekosystemie .NET.
6. **Testowanie jednostkowe:** Zapewnia narzędzia do tworzenia i uruchamiania testów jednostkowych, co pomaga w utrzymaniu wysokiej jakości kodu.
7. **Integracja z systemem kontroli wersji:** Umożliwia zarządzanie projektem za pomocą systemu kontroli wersji, takiego jak Git, dzięki wbudowanym funkcjom kontroli źródła.
8. **NuGet Package Manager:** Ułatwia zarządzanie zależnościami projektu poprzez integrację z menedżerem pakietów NuGet.

Aby korzystać z Visual C#, możesz pobrać i zainstalować środowisko programistyczne **Visual Studio**, które obejmuje edytor **Visual C#.**
Po zainstalowaniu możesz tworzyć, edytować i kompilować projekty w języku C# za pomocą tego narzędzia.

Proste przykłady programów w C#

1. Program "Hello World":

```
// Program "hello_world.cs":
using System;
class Program
{
   static void Main()
   {
      Console.WriteLine("Hello, World!");
      Console.ReadKey();
   }
}
```

Kompilacja **csc.exe** hello_world.cs
Np.
c:\WINDOWS\Microsoft.NET\Framework64\v4.0.30319**csc.exe** **hello_world.cs**
lub
c:\WINDOWS\Microsoft.NET\Framework\v4.0.30319\csc.exe hello_world.cs
wynik hello_world.exe

2. Program do dodawania dwóch liczb:

```
// suma_2liczb.cs
using System;
class Program
{
   static void Main()
   {
      Console.WriteLine("Podaj pierwszą liczbę:");
      int liczba1 = Convert.ToInt32(Console.ReadLine());
      Console.WriteLine("Podaj drugą liczbę:");
      int liczba2 = Convert.ToInt32(Console.ReadLine());
      int suma = liczba1 + liczba2;
      Console.WriteLine("Suma: " + suma);
   }
}
```

Kompilacja

c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\csc.exe suma_2liczb.cs

Wynik : suma_2liczb.exe

3. Prosty program sterujący:

```
using System;
class Program
{
   static void Main()
   {
      Console.WriteLine("Podaj swój wiek:");
      int wiek = Convert.ToInt32(Console.ReadLine());
      if (wiek >= 18)
      {
         Console.WriteLine("Jesteś pełnoletni. Możesz głosować.");
      }
      else
      {
         Console.WriteLine("Jesteś niepełnoletni. Nie możesz głosować.");
      }
   }
}
```

4. Program pętli (wypisywanie liczb od 1 do 10):

```
// petlal_1_do_10.cs
using System;

class Program
{
```

```
    static void Main()
    {
        for (int i = 1; i <= 10; i++)
        {
            Console.WriteLine(i);
        }
    }
}
```

5. Program obsługi tablicy:

```
// tablica_imion.cs
using System;

class Program
{
    static void Main()
    {
        string[] imiona = { "Anna", "Jan", "Katarzyna", "Piotr" };

        Console.WriteLine("Lista imion:");

        foreach (string imie in imiona)
        {
            Console.WriteLine(imie);
        }
    }
}
```

Te przykłady obejmują podstawowe konstrukcje języka C#,
takie jak deklaracje zmiennych, pętle, instrukcje warunkowe itp.

Zaleca się eksperymentowanie z tymi przykładami, wprowadzanie zmian i dodawanie własnych funkcji, aby lepiej zrozumieć język C# i jego możliwości.